CONTROL TRAFFIC COMPRESSION METHOD

The present invention is related to a compression method for RTCP traffic in media data transmission sessions. In particular, the method is intended to be employed in real-time or near real-time data packet transmission in an Internet Protocol (IP) network using a real-time protocol (RTP) for media data delivery and Real-time Control Protocol (RTCP) for controlling media delivery. Each of the protocols, RTP or RTCP, is allocated a fraction of the available session bandwidth according to the specifications given in RFC 1889.

The Real-time Transport Protocol (RTP), as defined in RFC 1889, is the de-facto standard that provides end to end network transport functions suitable for applications transmitting real-time data over multicast or unicast network services. It is augmented by the Real-time Control Protocol (RTCP) to allow monitoring the Quality of Service (QoS) of data delivery in a manner scalable to large multicast networks and to provide minimal control and identification functionality. RTP does not address resource reservation and does not guarantee Quality of Service (QoS) for real-time services.

RTP restricts the control traffic using two rules: First, it is recommended that 5% of the session bandwidth is allocated to RTCP traffic and it is shared by all participants in a session. Second, the minimum report interval for the transmission of feedback is recommended to be five seconds. All receivers in a session are using their own fraction with this 5% to calculate their report interval. The RTCP report interval `T` defines the time interval between two RTCP data packets from a source that has to be met. This interval depends to a large extent on the average RTCP packet size.

While these rules make RTP stable and usable for large multicast groups it is not optimal for unicast or small multicast scenarios. For the latter, more feedback per user would be beneficial and most likely possible. The problem was already identified and a new RTP profile RTP-AVPF is being standardised by the IETF's Audio Video Transport Working Group. With the new profile the recommended minimum feedback interval of five seconds is not applied. Therefore the receiver can send some early RTCP packets as feedback for packet losses, depending on the current session parameters.

IP based real-time multimedia application introduce a large Layer-3, Layer-4 and upper layer header overhead due to usually small payload sizes of single packets in a real-time data flow. Because of the restricted bandwidth of wireless links, header compression

represents an essential prerequisite for the mobile Internet, i.e., whenever an IP based mobile end device has to communicate with an IP based infrastructure. Robust Header Compression (ROHC) as defined in RFC 3095 is the state-of-the-art header compression scheme standardised by the IETF. It provides a complex framework that allows to fine tune compression efficiency versus robustness against link errors based on different link conditions. The protocol works by maintaining states at both end points of the first hop or last hop wireless link and by removing the redundancy of the packet headers and by encoding the information in an efficient way. The states of the compressor at the transmitting end (or endpoint) and the decompressor at the receiving end are also referred to as "context". The context contains relevant information from previous headers in the packet stream, such as static fields and possible reference values for compression and decompression. Additional information describing the packet stream may be also part of the context , for example information about how the IP Identifier field changes and the typical inter-packet increase in sequence numbers or timestamps.

Although in RFC 3095 there are four profiles for No Compression, RTP/UDP/IP, UDP/IP, and ESP, as well as one draft profile for IP only, there is no specification on how RTCP packets and headers can be handled using compression.

Video streaming over the mobile Internet as one of the key applications using RTP/RTCP is gaining the momentum. However, the lossy behaviour and long round trip time in wireless links makes the deployment of this kind of applications challenging enough. One reason are inter-frame video compression algorithms like MPEG-4 exploiting temporal correlations between the frame to achieve extremely high compression gain, but they also suffer from the well-known propagation of errors effect, since errors of a reference frame propagate to all the dependent different frames.

The object of the present invention is to optimise the bandwidth efficiency for RTCP traffic and the to reduce the RTCP report/feedback interval. The shared session bandwidth fraction for RTCP among all participants in a session and for bi-directional operation is limited. Spectrum efficiency is vital in sparse and expensive wireless links. Therefore how to use this limited bandwidth efficiently represents a need for applications deploy RTP via wireless links. The presented method aims at maximum exploiting bandwidth efficiency for RTCP traffic for these usages without exceeding the available RTCP bandwidth fraction. The RTCP report interval is the period between two consecutive reports from the same receiver for within the same session. It is affected by latency from two aspects. RTCP report latency is the period between a packet loss is

detected at the receiver and a report/feedback is sent, and the latency due the round-trip-time (RTT) of the link. While the latter is hard to avoid, the report latency can be exploited for optimisation. The formula for the calculation of this latency, which is defined as the report interval T, can be expressed as:

$$T = avg\_rtcp\_size * n / rtcp\_bw$$

For a typical scenario in which RTP/RTCP is used in unicast and small multicast sessions, the number of participants n is relatively fixed. To reduce latency at the cost of reducing the number of participants is not desired. This leaves only experimental space with the RTCP bandwidth fraction `rtcp_bw` and the average RTCP packet size `avg_rtcp_size`. As mentioned, approaches aim to reduce the report interval by increasing the RTCP bandwidth fraction, but they modify the rule of the RTCP bandwidth fraction being at maximum 5% of the total available session bandwidth. Also those approaches may encounter compatibility problems.

In view of the above discussions, it the only possibility to improve, that means to reduce the report interval of the RTCP protocol is to reduce the average RTCP packet size `avg_rtcp_size`. As the RTCP report interval T is directly proportional to the average RTCP packet size, compressing RTCP packets can reduce the packet size to a level up to 10% of the original RTCP packets. This will result in a smaller average packet size of control protocol packets and therefore in a much smaller report interval T.

Based on this, the present invention provides a compression method for RTCP traffic controlling a RTP media data transmission session. The compression principles described herein can be applied to basically any kind of link using RTP for real-time and near-real-time media delivery, in either wired/fixed networks or wireless/mobile networks.

The endpoints in a data transmission according to the present invention maintain the content (state) of the compressor and decompressor. Due to the structure maintained in the context, repairing and recovering of out-of-sync context at the decompressor is possible. Further, it is possible to dynamically define packet formats and the compressor's and decompressor's context.

The compression method initialises the context of the control traffic flow by initially transmitting context parameters to the receiving endpoint. If necessary, the context is updated during the session using smaller sized packets (compressed control packets). Latter packets are used in case a partial context update is performed. It is also possible

to update the context periodically using the initialisation packet. Context parameters can be categorized into static and dynamic parameters. Static context parameters are either a-priori known parameters or parameters not changing during a session. Dynamic context parameters, which are parameters changing during a session, are transmitted in newly defined packets or compressed control packets to the receiving end.

As the different possible context parameters (also comprising all fields of the standard RTCP data packets) are known, they can be advantageously categorised into static and dynamic context parameters. Based on this categorisation a header and data compression may be performed.

In order to further reduce the traffic overhead, a priori known context parameters can be omitted and have therefore not to be transmitted, though it is possible to perform compression of theses packets by using the compression and decompression mechanism described herein.

To initialise a session, at least one initialisation packet comprising these context parameters is transmitted to the receiving nodes. As the comprised parameters include static information, these information only have to be transmitted once. Therefore the total traffic volume to be transmitted can be significantly reduced. Dynamic context parameters are transmitted for example in control protocol specific packets (compressed control packets). Refresh packets allow the packet source to update context information at a receiving node. Control packets correspond mainly to those known from the standard RTCP protocol.

In contrast, compressed control packets are changed in their packet structure, such that their content's size (in bits) can be significantly reduced and new packets such as initialisation packets and refresh packets are introduced. Hence, the total average packet size of the control protocol's control packets can be significantly reduced in comparison to the standard RTCP protocol.

As the content of RTCP source description packets and RTCP bye packets is not frequently changing or does not occur often during a session, the corresponding source description packet and bye packet in the control protocol will not be compressed in the disclosed method, though it is possible to perform compression of theses packets by using the compression and decompression mechanism described herein. Both packets have a similar format as the corresponding RTCP packets.

After the context parameters are categorised, at least one initialisation packet is formed from these static context parameter and, if needed, from initialisation values for dynamic context parameters before they are transmitted. Refresh packets are formed from dynamic context parameters before the same are transmitted.

To reach a maximum level of compression, dynamic context parameters are further categorised into occasionally changing context parameters, context parameters of random character, counter-like context parameters, frequently changing context parameters and context parameters that regularly change by a fixed delta. Depending on the category of the dynamic context parameters, the parameters can be compressed by encoding to reduce their size before incorporating them into control data packets. Especially counter-like context parameters, frequently changing context parameters and context parameters that regularly change by a fixed delta can be encoded using least-significant-bit (LSB) encoding.

Employing least-significant-bit (LSB) encoding the K least significant bits of the encoded field value instead of the original field value are used, where K is a positive integer. After receiving K bits, a decompressor at the packet receiving end, which decompresses the compressed data packet, derives the original value using a previously received value as a reference.

Figure 1 shows the packet format of an initialisation packet used by the compression method to initialise a session,

Figure 2 shows the packet format of a refresh packet used by the compression method to update dynamic context parameters,

Figure 3 shows the packet format of a sender report packet used by the compression method,

Figure 4 shows the packet format of a receiver report packet used by the compression method and

Figure 5 shows the packet format of an application-defined packet used by the compression method.

To reduce the report interval `T`, the average packet size of the control protocol's data packets is reduced. The standard RTCP protocol, as defined in RFC 1889, uses the following packets to control a media data transmissions stream in a real-time or near real-time environment: Sender reports for transmitting and receiving static's from participants that are active senders in a media data transmission session, receiver

reports for receiving static's from participants that are not active senders, source description items for describing the sending source, bye packets for indicating the end of participation of a former participant and application-defined (APP) packets for transmitting applications specific data.

In order to reduce the size of the above-mentioned data packets, the fields in the packet structure are analysed first. Generally all fields in RTCP packets can be categorised in static context parameters, that are fields expected to be constant throughout the lifetime of the packet stream (session), and dynamic context parameters, that are fields that are expected to vary in some way, for example randomly, within a limited value set or range or in some other manner.

The dynamic context parameters (the dynamic RTCP packet fields) may be further categorised as follows: Occasionally changing context parameters, context parameters of random character, counter like context parameters, frequently changing context parameters and context parameters that regularly change by a fixed delta.

The occasionally changing context parameters are those fields occasionally change but revert to their original value after a limited number of packets. Regarding context parameters and field within standard RTCP packets, those value or fields are the reception report count (RC) that indicates the number of report blocks in the packet, the source count (SC) fields that indicate the number of synchronization sources or contributing sources in a source description packet or identifying the number of synchronization sources or contribution sources in a by packet, payload type (PT) fields that identify the individual packet type, source description (SDES) items comprising information to describe packet sources properties and sub type fields in application-defined (APP) packets allowing a set of application-defined (APP) packets to be defined under one unique name.

Those occasionally changing context parameters can be transmitted initially for initialisation but there should also be a way to transmit or update those fields if they change. Therefore the suggested control protocol with compressed data packets introduces a refresh packet, which is used to transmit context parameters for update purposes. The usage and structure of the packet will be discussed further down below.

Frequently changing context parameters comprise those parameters that are normally either constant or have values deducible from some other fields but that frequently diverge from this behaviour. Therefore, there must be an effective way to update the

frequently changing context parameter at the receivers or senders end. The mentioned refresh packets can be used in such a case or the respective fields are sent as they are in the newly defined control packets.

Fields that have to be frequently updated comprise the RTP time stamp, fields that are indicating the delay since the sender report received last (delay since last sender report), the time stamp of the last sender report, inter-arrival jitter fields that indicate an estimate of these statistical variance of the RTP data packet inter-arrival time and the length field of the RTCP packets.

A further category of dynamic context parameters are packets of random character. Examples for those parameters are the RTCP fraction loss in the bit map mask (BLP) of RTCP APP packet as specified by J. Ott et al. in "Extended RTP Profile for RTCP-based Feedback (RTP/AVPF)", Internet Draft, Oct. 2002. As those fields are completely random they are included as they are in all compressed packet headers.

The next category of dynamic context parameters are the counter-like context parameters. Those parameters are fields that behave like a counter with a fixed delta between the different counter values for all RTCP packets. The only requirement on the transmission encoding of those fields is that packet losses between the compressor on the senders end and the decompressor on the receivers end must be tolerable. If several of those fields exist, all those fields can also be communicated together. Such parameters can also used to interpret the values of frequently changing context parameters.

Examples for those fields are the RTP sequence number, the extended highest sequence number received field, the sender's packet count indicating the total number of RTP packets a sender has transmitted in the time frame between the beginning of the session and the generation of the packet comprising the sender's packet count, the packet sender's octet count that is indicating the total number of payload octets transmitted in RTP packets by the sender in the time frame between the beginning of the session and the generation of a packet comprising the sender's octet count and the cumulated number of packets lost, indicating the cumulated number of packets lost during transmission.

The last category of dynamic context parameters comprises those context parameters that regularly change by a fixed delta. Those fields usually increase by a fixed delta in succeeding packets. Thus, those fields are correlated with one another. In this context it

is advisable to initiate the field's value using an initialisation packet and then updating the field by transmitting their increment.

An example for a context parameter that regularly changes by a fixed delta is the RTP time stamp.

Further, a third category besides the static and the dynamic context parameters can be determined. So-called well-known or a priori known fields within the standard RTCP packets may be either transmitted during initialisation or are omitted. An example for an a-priori known context parameter is the RTCP version field.

The described categorisation of context parameters can be, for example, performed by referencing tables used for packet and header compression. It would also be possible to dynamically categorise the context parameters within the suggested new RTCP compression method.

After the context parameters have been categorised, a part of the dynamic context parameter is encoded to reduce their size. In particular, counter like context parameters frequently changing context parameters and context parameters that regularly change by a fixed delta are least-significant-bit (LSB) encoded such that the original field size (in bits) may be substantially reduced.

After categorising the context parameters the control protocol's packets are formed. To initialise a session an initialisation packet is formed and transmitted. The packet format of an initialisation packet is shown in figure 1. The packet contains static context parameters such as a padding flag, a synchronization source of the sender and the source, and a contributing source field in the "static chain" field of the packet. The "static chain" field is thereby variable in its length as well as the "dynamic chain" field of the packet. Also incorporated in the initialisation packet are the source count and reception report count, a payload type identification, one or more SDES items and a subtype field for application-defined (APP) packets.

The occasionally context parameters can be also integrated in the formed initialisation packet using their initial value. These initialisation values of the occasionally changing context parameters are located in the "dynamic chain" field of the initialisation packet.

Once the occasionally changing parameters are initialised, they can be updated in the following by refresh packets.

In detail the compressed initialisation packet comprises a context ID (CID, "`Add-CID octet`") that identifies the state of the decompressor to be used at the packet receiving end to decompress the initialisation packet at the beginning of the packet, a packet identifier ("`1111110D`") to enable the packet receiver to identify the packet type, profile information ("`Profile`") for the sender's profile, cyclic redundancy check field ("`CRC`") for checking data integrity of the initialisation packet, a static information chain ("`Static Chain`") comprising static context parameters and finally dynamic information chain ("`Dynamic Chain`") comprising dynamic context parameters, that have to be initialised once. The latter correspond to the before-mentioned occasionally changing context parameters, such as the source count, the reception report count, the RTCP payload type, SDES items and the subtype field for application-defined (APP) packets.

Figure 2 shows the packet format of a refresh packet. As the latter mentioned fields of the initialisation packet are dynamic, the new refresh packet is introduced to update those fields. In detail the refresh packet comprises a context ID (CID, "`Add-CID octet`") for identifying the state of the header-decompressor to be used at the packet receiving end to decompress the refresh packet, a packet identifier ("`11111000`"), profile information ("`Profile`") of the packet sender, a cyclic redundancy check ("`CRC`") field for checking data integrity of the refreshing packet and a dynamic information chain ("`Dynamic Chain`") comprising the dynamic context parameters that have to be updated.

Additionally the initialisation packet and the refresh packet may comprise up to two additional bytes following the packet identifier in case large context identifiers (CID) are used.

Figures 3 and 4 show a compressed version of the sender and receiver report packets and figure 5 shows a new compressed application-defined (APP) packet.

The source description packets and the bye packets correspond to the standard packet format as suggested in RFC 1889. This is because those packets occur very rarely during a session such that they are compression would not reduce the average packet size of RTCP packets significantly.

The sender report packet, as shown in figure 3, comprises a packet header and at least a report block. The report block/s can be followed by profile-specific extensions. In the profile-specific extensions all fields that fall into one of the above mentioned categories

can be also compressed using least-significant-bit encoding. Hence, the extension fields' size can be also reduced leading to a smaller packet size on average.

The abbreviation "LSB" in the figures stands for "Least Significant Bit" and indicates that the respective fields are least-significant-bit encoded.

The header of the sender report packet comprises a packet identifier ("`111`") to identify the sender report packet type. Further, a reception report count field ("`RC`") is indicating the number of report blocks comprised in the compressed sender report packet. An active sender flag ("`S`") indicates whether participant that forms the report block is an active sender [Gu1]or not[FH2]. The cyclic redundancy check ("`CRC`") field is used to check data integrity of the compressed sender report packet. A padding flag or bit ("`P`") is indicating whether the sender report packet contains an additional padding field at the end of the packet. The additional padding field is not a part of the actual context parameters.

A least-significant-bit (LSB) encoded RTP time stamp ("`LSB Scaled RTP Timestamp`") is further comprised in the header. An extension flag ("`X`") is indicating whether the packet comprises profile-specific extensions in a special extension field at the end of the packet.

To further reduce the sender report packet size, the sender's packet count field is also least-significant-bit (LSB) encoded. The sender's packet count field ("`LSB Sender's Packet Count`") in the header of the sender report packet indicates the total number of RTP packets the sender has transmitted in the time frame between the beginning of the media data transmission session and the generation of the respective sender report packet.

Further, the header of the sender report packet comprises a field for the sender octet count indicating the total number of payload octets transmitted in RTP data packets by the sender in the time frame between the beginning of the session and the generation of the sender report packet. Again, the field is least-significant-bit (LSB) encoded to reduce the size of the packet header. In the figure, this field is split into two parts ("`LSB Sender Octet Count Part1`" and "`LSB SOC P2`"), the 5th byte of the packet and the five first bits of the 6th byte of the packet contain the sender's octet count.

The packet header further comprises a field for indicating the sender report's length ("`LSB Len SR`"). This field is also least-significant-bit (LSB) encoded. The end of the header is marked by the "`+=+=+=`" line after the 6th byte of the packet.

The at least one report block, in the sender report packet comprises the following fields:

A fraction lost field ("`fraction lost`") that indicates the number of packets lost divided by the number of packet accepted to be received, a cumulated loss field ("`cummu. loss`") indicates the cumulated number of packets lost during transmission. To reduce size, the cumulated loss field is least-significant-bit (LSB) encoded as the remaining fields of the report block. A sequence number cycle field ("`LSB SN Cycle`") is indicating the sequence number cycle of the extended highest sequence number of received packets. The highest sequence number field ("`LSB Highest SN`") indicates the highest sequence number received by the sender of the sender report packet. The inter-arrival jitter field ("`intera. jitter`") comprises an estimation value of the statistical variance of the RTP data packet inter-arrival time. Further included in the report block is an RTP time stamp field ("`LSB TS last SR`") indicating the time since the last sender report has been sent. A field ("`LSB DLS`") for indicating the delay since the last compressed RTCP sender report is also included.

All fields in the report block except the fraction lost field, are encoded using least-significant-bit (LSB) encoding.

A single report block is four bytes long (bytes seven to ten shown in the figure). As indicated in the figure, multiple report blocks may be comprised by a single sender report.

Besides the sender report packet, a compressed version of the RTCP receiver report packet is suggested in the following. The receiver report packet, as shown in figure 4, comprises a header (bytes one to three) and at least one report block, which is similar to the report block described before. The compressed receiver report packet as well as the compressed sender report packet may also comprise profile-specific extensions at their end, indicated by an extension flag ("`X`") in the packet header.

The header of the receiver report packet comprises a packet identifier ("`111`") to identify the receiver report packet type thus that the receiving end may recognize the compressed version of the receiver report. A reception count field ("`RC`") indicates the

number of report block comprised in the receiver report packet. As the sender report packet, a [Gu3]receiver report packet may comprise several report blocks following the respective packet header.

An active sender indication flag ("S") indicates the status of the session participant who generated the respective report block. Further, a cyclic redundancy check field is included to verify data integrity. A padding flag ("P") is indicating whether the receiver report packet contains an additional padding field at the end of the receiver report packet. The additional padding field is not part of the actual context parameters. Lastly, a length field ("LSB Length RR") is included in the header of the receiver report packet to indicate the lengths of the compressed RTCP receiver report in least-significant-bit (LSB) encoded format.

Finally, an application-defined (APP) packet format is shown in figure 5. The user of the suggested compressed application-defined (APP) packet format only applies in case the enhanced RTCP feedback as suggested by Ott et al. is used. Therefore, the packet specification is rather application-specific - as its name indicates - than a general approach.

The compressed application-defined (APP) packet format comprises a packet identifier ("111") for identifying the compressed application-defined (APP) packet type. A feedback type field ("FMT") is indicating the feedback type provided in this packet. Further, the length of the packet is indicated in the feedback length field ("LSB Feedback Length"). This field is least-significant-bit (LSB) encoded. A bitmask field ("BLP") is indicated last packets. The first bit is the BLP field (bitmask field) allows reporting loss to any of the RTP packet immediately following an RTP packet indicated by a packet identifier. In case the feedback type field (FMT) is indicating a [Gu4]generic acknowledgement, the first bit of the BLP field is (the so-called R bit) is 1. In this case the BLP field is used to identify the number of additional packets that are acknowledged by the compressed application-defined (APP) packet. Otherwise, if R = 0 the BLP field carries a bitmask indicating lost packets.

In summary, the above suggested compressed control packets as well as the two newly introduced packet formats (initialisation packet and refresh packet) are intended to reduce the overall average packet size of control protocol's packets, allowing to reduce the report interval T.

On [Gu5]one hand, the data volume is reduced by sending static context parameters in the initialisation packets "in advance" as well as to initialise occasionally changing context parameters. In order to be able to update an occasionally changing context parameter in case it is changing, the refresh packets are used to do so. On the other hand, most of the control packet fields are encoded, such that their size is further reduced.

Thus, it is possible to reduce the average size of the compressed control data packets in comparison to the standard RTCP protocol. Hence, with the suggested packet formats it is possible to significantly reduce the report interval T, without extending the allocated bandwidth for control traffic[Gu6][FH7]. Consequently, by being able to provide feedback in shorter time intervals, the participants of a media data transmission session can adapt to changing transmission environments faster than in sessions using the standard RTP/RTCP protocol combination. Hence, the overall quality of the transmitted (or broadcasted) application data, such as MPEG-4 encoded video data, can be significantly improved.

The suggested header and data compression mechanisms deal only with the RTCP header and data part and not the lower layer UDP/IP headers. Therefore, compared to header compression scheme like suggested in RPF 3095, which is generally applicable to the last hop or first hop point-to-point link, the approach described herein can be applied end-to-end. The intermediate hops along the way to a packet's do not need to care about the compressed control packets, as they see them either as Layer-3 IP packet data or as Layer-4 transport layer packet data. No additional overhead for the intermediate hosts will be introduced. However, if used together with Robust Header Compression for lower layers' headers in first/last hop wireless links, even more bandwidth can be saved.